A Model-Based Admission Control for 802.11e EDCA using Delay Predictions

A. Bai, T. Skeie, P. E. Engelstad

Abstract - This paper presents a unique approach for a model-based admission control algorithm for the IEEE 802.11e Enhanced Distributed Channel Access (EDCA) standard. The analytical model used as the foundation for the algorithm covers both non-saturation and saturation conditions. This allows us to keep the system out of saturation by monitoring several variables. Since the medium access delay represents the service time of the system, it is used as the threshold condition to ensure that the queuing delay is within reasonable bounds. The paper describes the admission control algorithm and several simulation results are presented and discussed.

Keywords: 802.11e, admission control, analytical model, delay, saturation.

I. INTRODUCTION

IEEE 802.11 WLAN [1] is the most widely used technology for wireless access to wired Local Area Network (LAN) infrastructures and to the Internet. To meet the demands for Quality of Service (QoS), the new IEEE 802.11e amendment [2] to the standard was developed, and it has recently been accepted as a new standard. IEEE 802.11e includes the Enhanced Distributed Channel Access (EDCA), which provides differentiation between four different priority classes – called Access Categories (AC).

Normally, traffic differentiation alone is not sufficient to accommodate appropriate levels of QoS. In addition, some sort of admission control mechanism is needed to enforce that the users get their agreed QoS. The mechanism also ensures that traffic not authorized for network resources is not destroying the quality of the admitted traffic by consuming network resources. The IEEE 802.11e [2] standard does not specify how admission control should be implemented, and this is left to the implementer. Developing a model-based admission control for IEEE 802.11e EDCA is the objective of this paper.

Admission control for IEEE 802.11e has lately gained a lot of interest in academia, where several algorithms possessing interesting features have been proposed [3-12]. Most algorithms assume that the system is in saturation i.e. that all stations always have frames to transmit.

The problem with this assumption is that when the system is in saturation, the transmission queues grows to infinite lengths (theoretically) or results in massive queue drops when the buffer space is finite (real-life). In both cases, the excessive queuing delay and queue drop makes it hard for upper-layer protocols to communicate. In fact, the system must be in a non-saturation state if meaningful communication shall be possible. In this paper, we therefore argue that an admission control algorithm must ensure that the system is in non-saturation, so that the traffic classes (at least those of the highest priority) have limited queuing lengths.

There are two main approaches for admission control in 802.11e; measurement-based and model-based admission control [13].

Measurement-based admission control algorithms use measurement of some network parameters to decide if a flow should be accepted [3-7]. However, most of them assume that the network is fully saturated with traffic, i.e. that all stations always have frames to transmit. Hence, they are all imperfect for the non-saturation scenarios where the transmission queues might be temporarily empty. However, there are some measurement-based algorithms that do not assume saturation conditions. The work in [7], for example, presents an ATL (Admitted Time Limit) algorithm that is dynamic. Here, the saturation assumption is discarded, and the algorithm therefore performs better for non-saturation scenarios.

The model-based admission control approach deploys an analytical Markov model (Bianchi model) to determine if flows should be admitted or not [9-12]. Also here, the proposals use models that assume saturation conditions. They are therefore not adequate if the objective of the admission control algorithm is to ensure that some of the traffic classes are not saturated.

One reason that previous proposals might have assumed saturation conditions is that few non-saturation

Bianchi models of IEEE 802.11e EDCA are available. Recently, however, a number of proposals have emerged (e.g. [14-19])

Another recent achievement of such analyses is that the delay of the system has been expressed in terms of the z-transform [20]. This makes it easier to provide a full description of the delay [21], and to predict the queuing delay [22]. An important result from this work is that the utilization factor, ρ_i of AC *i*, can be written [21, 22]:

$$\rho_i = \min(1, \overline{\lambda_i D_i^{SAT}}), \quad (1)$$

where λ_i is the traffic intensity and $\overline{D_i^{SAT}}$ represents the mean medium access delay with the post-backoff delay included. Note that the queue length is in theory infinite when $\rho_i = 1$, while an AC is not in saturation if $\rho_i < 1$.

In this paper a model-based admission control for IEEE 802.11e is proposed, which is not only based on saturation conditions. Indeed, the basic idea behind our delay oriented admission control is to ensure that the utilization factor $\rho_i < 1$ for the admitted traffic.

To the best of our knowledge, no one has presented an admission control for 802.11e EDCA, which is based on delay instead of throughput, although the idea was discussed briefly in [21].

The next section gives an overview on how to carry out measurements that make the analytical model applicable to our algorithm. In Section III the benefits of using delay instead of throughput as the threshold are demonstrated. Our admission control algorithm is explained in Section IV, before the solution is validated in Section V. Finally, the conclusions are drawn in Section VI.

II. Linking Measurements to an Analytical Model

In order to monitor the utilization factor, ρ_i, our admission control algorithm needs to measure the transmission probability τ_i, which is defined as the probability for a transmission in a generic timeslot, referred to as a "Markov slot" in this paper. By measuring τ_i, it is possible to use an analytical model to calculate $\overline{D_i^{SAT}}$. (For a more comprehensive background of the equations used to calculate this, refer to [21, 22]). When the traffic intensity is known at all times, it is possible to calculate ρ_i according to Eq. (1).

In many models, including that in [21], all properties of the model can be expressed in terms of τ_i. Thus, in order to incorporate the needed equations, τ_i needs to be measured at run time. However, a Markov slot does not have a static length in the analytical Markov model (e.g. of Engelstad and Østerbø), but varies according to the situation. An empty Markov slot corresponds to a timeslots of IEEE 802.11 with a duration of 20 μs. A busy Markov slot, on the contrary, may contain a successfully transmitted frame. In this case, the Markov slot spans over the time to transmit the data, SIFS (Short Inter-Frame Space) time, time to send the ACK packet and the DIFS (Distributed Coordination Function Inter-Frame Space) time, as well as the almost negligible propagation delay. Several Markov timeslots are indicated in Figure 1, where there are first several empty Markov slots, before a transmission Markov slot.

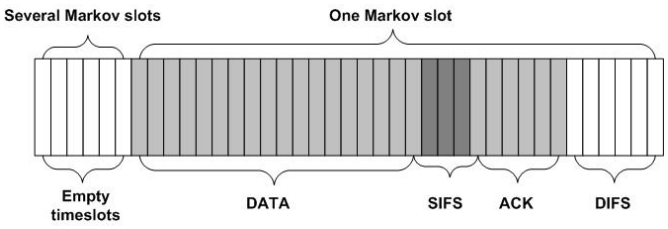

Figure 1. Empty Markov slots and a busy Markov slot containing a successfully transmitted frame

If there is a collision, EIFS (Extended Inter-Frame Space) time is included as shown in Figure 2.

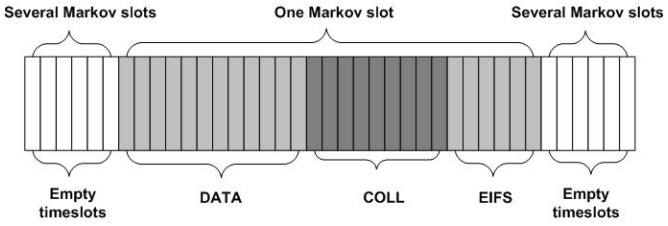

Figure 2. Empty Markov slots and a Markov slot containing a collision

We have implemented the measurement of τ_i and our admission control algorithm in ns-2 [23]. We have also used the TKN module [24], which implements the IEEE 802.11e standard, and expanded it to include our model-based algorithm.

In our implementation, three different counters are kept: $trans_i$, $Markov\_slots$ and $coll_i$. Trans is a counter that keeps track of every successful transmission for a priority, coll keeps track of every collision that occurs for a given priority and

Markov_slots counts the number of Markov slots (including idle timeslots, successful and unsuccessful transmissions) within a beacon period. A beacon frame is sent by the QAP (Quality of Service Access Point) at regular intervals, and the frame contains various parameters like the EDCA parameters.

τ_i is calculated based on the $trans_i$, $Markov\_slots$ and $coll_i$, counters, and updated at every beacon period as follows:

$$\tau_i \leftarrow f * \tau_i + (1-f)\frac{trans_i + coll_i}{Markov\_slots} \qquad (2)$$

Here, f is the damping factor, which is set to 0.9 in our scenarios.

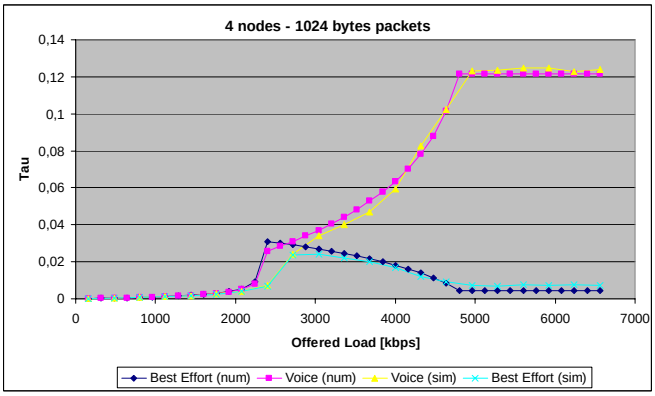

Figure 3. Development of τ_i

Figure 3 shows the development of τ_i for a scenario with 4 nodes. Here, the measurements of τ_i are compared with the corresponding values calculated by the analytical model under the given traffic intensity. Each node is sending two flows, one with voice and one with best effort. The system reaches saturation when the offered load is close to 5000 kb/s, and τ_i has stabilized. The detailed settings for the scenario are found in Table 1, in Section V.

The analytical results from the equations of [21, 22] are plotted in Figure 3 as well. It is easy to see that the simulation results of τ_i matches the numeric results of the analytical model. This shows that simulation and theory matches quite well. The numeric results of the analytical model were found by Mathematica.

The figure demonstrates, by using the measurement method described above, that it is possible to measure τ_i with satisfactory accuracy. Hence, τ_i can easily be incorporated directly into the equations of the analytical model, and other performance characteristics of the real system can be calculated.

As explained above, the goal of the admission control algorithm is to stay out of saturation; that is to keep $\rho_i < 1$. The development of ρ_i is shown in Figure 4, given the same scenario as in Figure 3. When the offered load exceeds approximately 4300 kb/s, Figure 3 and 4 show a rapid increase in τ_i and ρ_i. Here, ρ_i is calculated from the τ_i values presented in Figure 3. The "num" curves for ρ_i corresponds to the τ_i in Figure 3 that were calculated numerically from the analytical model. The "sim" curves for ρ_i corresponds to the τ_i in Figure 3 that were measured real-time in the simulator.

In Figure 4 it is observed that when the voice class reaches saturation at around 5000 kb/s, ρ_i crosses the line $y = 1$, which further supports the theory of keeping $\rho_i < 1$.

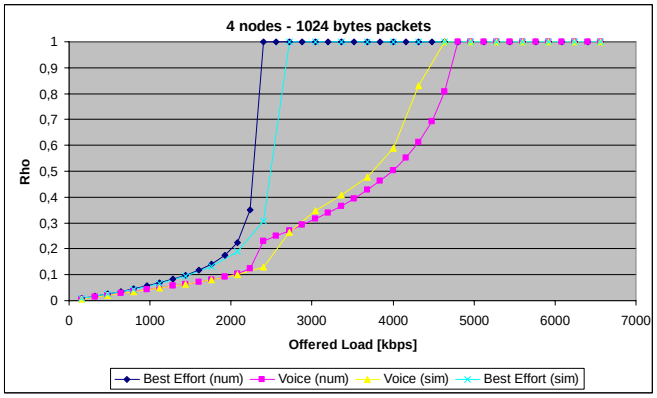

Figure 4. Development of ρ_i

III. Delay versus Throughput

An admission control algorithm that deploys delay predictions as the threshold for accepting or rejecting new flows, will in theory achieve better utilization of the channel compared to other admission controls that only considers throughout as the threshold. This is because throughput starvation occurs after queuing starvation (i.e. after queue overflow). This was also the conclusion of Engelstad and Østerbø [21, 22], and it is quite straightforward to demonstrate it with a simple scenario.

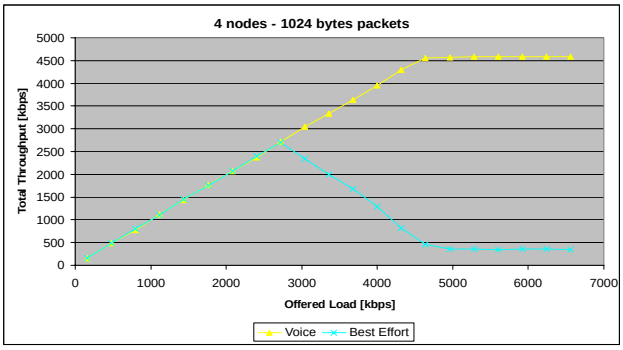

Figure 5. Total throughput

Figure 5 shows the total throughput for the scenario given in section II. The behaviour of the flows is as expected [2], where voice class takes bandwidth from the best effort class. The figure shows that when the offered load is close to 5000 kb/s, the system is in full saturation, and voice's throughput stabilizes at around 4500 kb/s.

However, in Figure 6, where the delay for the voice class is shown, a different view of the same scenario is plotted. Already when the offered load exceeds 4300 kb/s, the delay for the voice class starts to increase very fast. When the voice class reaches maximum throughput at around 5000 kb/s, the delay for the voice class is already very high.

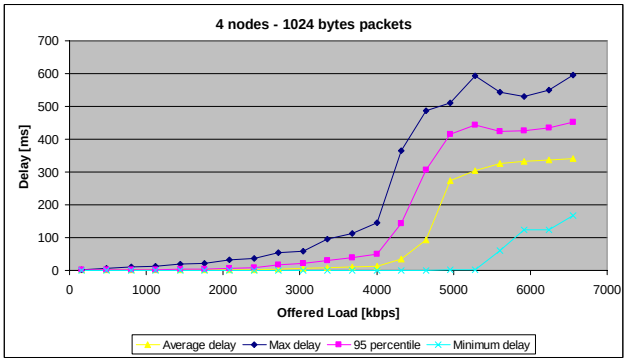

Figure 6. Delay for the voice class

So in order to gain approximately 500-600 kb/s more in throughput for this scenario, the cost is about 300 ms in the average delay. An admission control which considers delay and stops the system from reaching saturation will thus achieve better delay results without scarifying too much of the throughput.

We have therefore developed an admission control algorithm that is based on queuing starvation instead of throughput starvation. The next sections explain our algorithm in details.

IV. Model Based Algorithm Using Delay Predictions

It is not enough to only measure τ_i at run time, using Eq. (2). The admission control algorithm must also be able to predict when the system will be in saturation before the system is actually in saturation. The algorithm can then reject flows that will put the system in saturation. How to predict when the system will be in saturation based on one or more parameters is not straightforward, but can be accomplished by several approaches.

We have solved this by splitting the admission control decision process into two parts; Phase 1 and Phase 2. In Phase 1 several preliminary calculations are conducted before deciding if a flow is either rejected or passed on to Phase 2. In Phase 2 the flows are monitored for a given time period, before a final decision is made. Figure 7 illustrates the flow of our delay based algorithm.

Every station that implements our admission control algorithm must be Phase 1 aware. This is because when a station wants to start a new flow, it must send an ADDTS (Add Traffic Stream) frame that contains a TSPEC (Traffic Specification) with the requirements of the flow [2]. The QAP will use the requirements of the flow for its pre- calculations in Phase 1. Finally, the QAP has to answer every request with an ADDTS, telling the station if the flow was rejected or accepted [2].

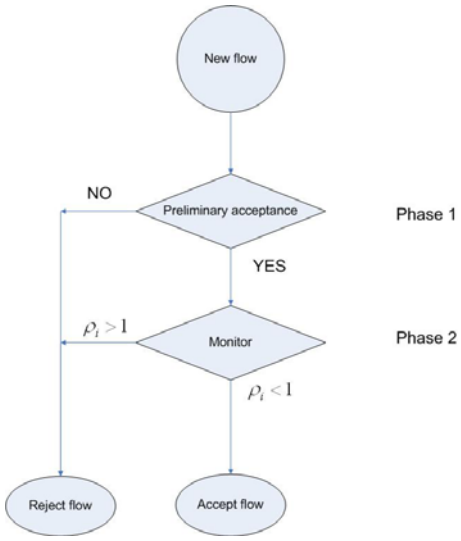

Figure 7. Flow of model-based admission control algorithm

In our implementation the stations are aware of Phase2, although, this is not a requirement. Every flow that is rejected during Phase 2, will receive a DELTS (Delete Traffic Steam), which informs the stations to

tear down the flow, according to the standard's suggestions [2]. (A detailed description of ADDTS, DELTS and TSPEC frames can be found in the IEEE 802.11e standard [2].)

The monitoring phase (Phase 2) is a critical part of the algorithm, because if the algorithm is configured incorrectly, the quality for the already admitted flows can be aggravated. Already admitted flows should not be affected by new flows, not even those who are requesting acceptance. Therefore, two variables are very important for this critical phase; how long the monitoring phase in Phase 2 lasts and the threshold of ρ_i for Phase 1.

The threshold of ρ_i for Phase 1 should be lower than the threshold of ρ_i for Phase 2. This is because the functions used to predict τ_i are not 100% accurate and therefore some error fluctuation must be considered. The effects of different ρ_i thresholds for Phase 1 and 2 are shown in Section V.

If the threshold for Phase 1 is too high, a new flow can aggravate the quality of admitted flows when it enters Phase 2. For example, a flow that was accepted in Phase 1 can turn out to use too much bandwidth in Phase 2, and thus push the system into saturation. The flow is rejected quickly if the system exceeds the ρ_i threshold in Phase 2, but the damage could already be done. This is why the ρ_i threshold in Phase 1 should be lower than the ρ_i threshold in Phase 2.

The length of the monitoring phase also plays an important role. With a too short monitoring phase, flows that eventually would push the system beyond the ρ_i threshold might be accepted. On the other hand, a too long monitoring phase might waste resources. We have had good experience with a monitoring phase of 30 beacon periods (Table 1).

A. Phase1

In Phase 1, some predictions are made regarding the quality of the system if a new flow is accepted. These predications are based on one or more assumptions regarding the properties of the system, such as the EDCA parameters, the behavior of the nodes or even the trend of the transmission probabilities based on regressions equations.

The intention of this phase is to see if the usability of the system will be considerably aggravated if a new flow is accepted. If the pre-calculations show that a new flow with a specific set of requirements will, with a high probability, aggravate the quality of the system, then the flow will be rejected before Phase 2.

When a flow is rejected during Phase 2, due to severely aggregation of the usability of the system, it means that both time and resources have been wasted. Ideally, it had better be rejected already in Phase 1. The intention of Phase 1 is simply to save resources and time by eliminating flows that will never be accepted in any case.

There are several possible algorithms that can be used in Phase 1, but no matter which scheme is chosen, some assumptions must be made. This is because we can not predict the future, but can only simulate or guess what might happen. One possible scheme is to i.e. calculate how many timeslots a transmission would take. Then it is easy to calculate a new τ_i at run time, and compute the corresponding ρ_i. However, it is difficult to predict the correct number of timeslots a certain offered load will consume, because it depends on a large number of other system parameters. For instance, it depends on how many transmissions are already going on, the EDCA parameters used, how many stations are actively transmitting and so on.

Another possible algorithm is to simulate a specific scenario and extract the τ_i values when varying the offered load from zero until saturation is reached. Then it is possible to calculate several regression equations for each of the different τ_i. These regression equations can be used to predict τ_i based on a given offered load with a relatively high probability. However, this assumes that the system that applies the regression equations must use the same settings as the system that the regression equations were extracted from. Both the EDCA parameters and the number of actively transmitting stations must be the same.

It is of course possible to have several different regression equations based on different EDCA parameters and different sets of actively transmitting stations.

In our implementation, we have chosen to use this scheme because it is easy and fast to implement and gives a good probability of τ_i if the assumptions are met.

In order to find the regression equations, several simulations were done with the settings listed in Table 1. The average of all simulations (32 samples) is plotted in figure 8. Priority 0 (voice) and 2 (best effort) of 802.11e were used, together with the same number of nodes as the scenario in Section II. Given the graphs in Figure 8, it is straightforward to find the regression equations.

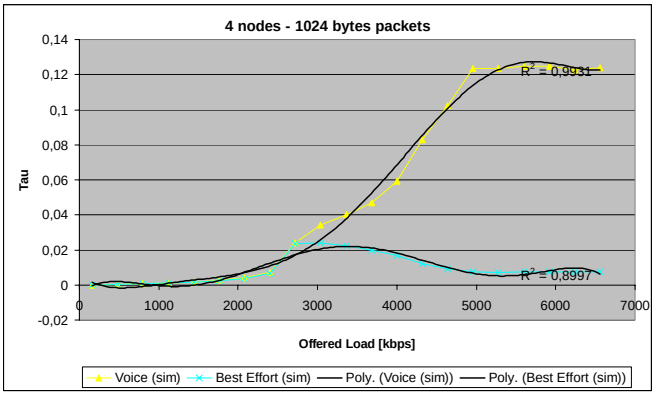

Figure 8. Regression lines for voice and best effort

From Figure 8, two polynomial regression equations were extracted, which are shown in Eq. (3) and (4), using the Excel Trendline function [25]. Both equations have a very high number of decimals, since it is important that the equations return τ_i with a high degree of accuracy. In Figure 8, the R^2 values are also plotted, which indicate how close the regression line is to the actual τ_i value. It is also important to remember that the equations should be used only within the boundaries of the regression lines, which is from zero to 6500 in our case.

$$\tau_0(x) = 9{,}65457765909510 * 10^{-23} x^6 - 1{,}85636259996843 * 10^{-18} x^5 + 1{,}28219399723190 * 10^{-14} x^4 - 3{,}91286873471391 * 10^{-11} x^3 + 5{,}83763592297765 * 10^{-8} x^2 - 3{,}60344390321835 * 10^{-5} x + 5{,}96900387315322 * 10^{-3} \quad (3)$$

$$\tau_2(x) = -1{,}03497427901352 * 10^{-22} x^6 + 2{,}06193053962155 * 10^{-18} x^5 - 1{,}51527601173359 * 10^{-14} x^4 + 4{,}96792882244998 * 10^{-11} x^3 - 6{,}97297326427663 * 10^{-8} x^2 + 3{,}77322568734752 * 10^{-5} x - 4{,}87961670746017 * 10^{-3} \quad (4)$$

The regression lines extracted from the scenario have R^2 values of 0.9931 and 0.8997. The regression equation for the voice class has the closest match, while the regression line for best effort is a little off at some points. The values returned from the regression equations should therefore not be taken as guarantees, but merely as indications. Even so, they return a τ_i with a relatively high accuracy.

When a node sends a ADDTS with a TSPEC containing the requirements of a new flow, the QAP finds a probable τ_i, based on the regression equations, and calculates a new ρ_i based on the total offered load and τ_i. The total offered load is the sum of the bandwidth requested from all the flows (each flow has to specify the wanted bandwidth in the TSPEC).

The new ρ_i is then evaluated against a threshold, α_{Phase1}, which is not necessarily equal to the threshold used in Phase 2. If $\rho_i < \alpha_{Phase1}$, the new flow is passed on to Phase 2. Otherwise, it is rejected.

The intention of Phase 1 is to show a simple, but flexible algorithm. It is, however, the combination of Phase 1 and Phase 2, using the delay predictions, that is the main contribution of this paper. Therefore, a simple Phase 1 was chosen, in order to emphasize the general idea of our concept, and not Phase 1 in particular.

B. Phase 2

The goal of this phase is simply to verify the conclusion that was drawn in Phase 1. If a flow has passed Phase 1, it is highly probable that the usability of the system can tolerate the new flow. However, it must be absolutely certain that this conclusion is correct. The only way to confirm this, is to monitor the actually effect a new flow has on the system. A flow that has passed Phase 1, is marked as temporarily accepted by the QAP.

Therefore, when a flow enters Phase 2, it will start transmitting data as if it already has been fully accepted.

However, the QAP measures and evaluates the flow after every beacon period. If the system at any stage in the monitoring phase will be aggravated beyond toleration, so that $\rho_i > \alpha_{Phase2}$, the new flow is rejected.

After a predefined number of beacon periods, the QAP marks the flow as fully accepted if the usability of the system is still satisfied and treats it like all the other fully accepted flows. The way of categorizing schemes as temporarily admitted and fully admitted, can be enhanced in many ways. This issue, however, is beyond the scope of this paper.

It is usually desirable to keep ρ_i below a certain threshold and not always use 1 as the limit. Thus, we normally use $\alpha_{Phase1} \le \alpha_{Phase2} \le 1$. For instance, in the scenario from section two, a configuration of $\alpha_{Phase2} = 0.6$ would guarantee that the 95 percentile for the delay would be below 100 ms. It is possible to have different α_{Phase2} for different priorities.

V. SIMULATIONS

For our simulations we used the same scenario as outlined in Section II. There are four nodes that are

actively transmitting flows of two different Access Categories: voice (AC_VO) and best effort (AC_BE). Each node can send as many flows as it wants, since it is the total offered load that is used as input to the regression equations. The detailed settings used for the simulations are listed in Table 1.

Parameter	Value
Channel Rate	11 Mbps
Packet size	Voice (AC_0) = 1024 Bytes Video (AC_2) = 1024 Bytes
Traffic generator	All nodes sending CBR traffic
Link delay	1us
Stabilize time	50s
Simulation duration	180s
Queue length	50 packets
Physical layer settings	802.11b
EDCA parameters [AIFS, CWmin, CWmax, TXOP limit]	Voice (AC_0) = [2, 7, 15, 1.5] Best Effort (AC_2) = [3, 31, 1023, 1.5]
Number of stations	4
Admisson control	Enabled
Beacon interval	0.1 seconds
Monitoring beacons	30

Table 1. Detailed configuration for the scenario

Each station has ten flows that it will try to request admission for; five flows require throughput of 1.5 Mb/s each and five flows require throughput of 300 kbps each. The five flows requesting 1.5 Mb/s are started first, and there is an interval between each flow at 15 seconds. This is simply to make it easier to see when a flow is admitted or rejected. The flows requiring 300 kb/s follow the same procedure, and the first 300 kb/s flow starts after the last 1.5 Mb/s flow. The nodes will try to request admission for all ten flows of both the voice and best effort classes.

Each scenario was executed 32 times, each time with different seed values, in order to gain statistical significance in the results. The results presented here is the average of all the simulations.

As stated before, because of a relative simple Phase 1, the simulations are somewhat limited. We underline that Phase 1 could be replaced by a more complex algorithm. However, because of limited space, we have targeted a small and compact Phase 1 algorithm, and therefore our simulation scenario may be limited. Again, it is the general algorithm with the delay predictions that is the real strength of our concept.

Figure 9 shows the throughput results from the scenario for a single station. The first 1.5 Mb/s flow is accepted for both voice and best effort as shown in the figure. However, the second 1.5 Mb/s flow is rejected by the best effort flow while the voice class accepts it. After that, all the remaining 1.5 Mb/s flows are rejected.

After 75 seconds, the first 300 kb/s flow tries to transmit, but only the voice class accepts the flow. The voice class also accepts the next 300 kb/s flow, before it rejects the rest. The best effort class rejects all the 300 kb/s flows.

In the scenario illustrated in Figure 9, a α_{Phase1} of 0.4 for the best effort class was used. According to Figure 3, a value of 0.4 for α_{Phase1} would indicate that the best effort class could support traffic up to about 2 Mb/s. Therefore, it is not unreasonable that the best effort class only admits 1.5 Mb/s. It did not accept another 300 kb/s flow because the channel quality introduced a higher τ_i (for the already admitted flows) than expected. The voice class has a higher α_{Phase1} of 0.5, and can accept up to 4 Mb/s according to Figure 3. Figure 9 shows that the voice class admitted up to 3.5 Mb/s, which is also below what Figure 3 indicates. This is also because τ_i of the already admitted flows is higher than expected during run-time.

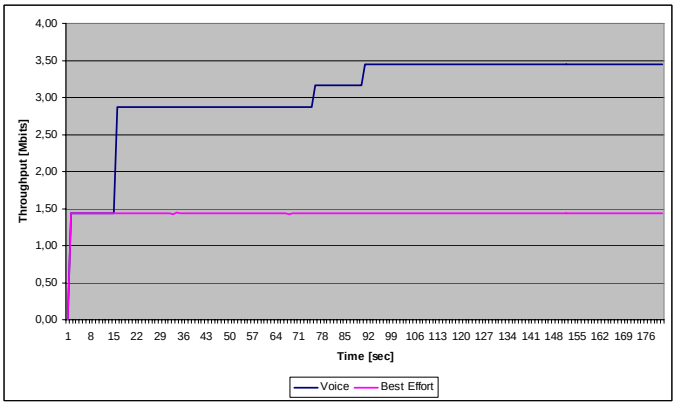

Figure 9. Throughput results for voice and best effort

Figure 9 shows very stabile throughput for both the voice and best effort class, but as shown in Section III, it does not necessarily mean that delay will not suffer. Figure 10 shows the average delay results for each flow in the voice class. As could be expected, the delay is quite low, with an average delay of 2 *ms* when the throughput is at 3.5 Mb/s. This is clearly acceptable, and in line with Figure 3.

The delay results for the best effort class are also in line with what could be expected and will therefore not be shown. The average delay for the best effort class was also around 2 ms, and the maximum delay was 21 *ms* in the worst case.

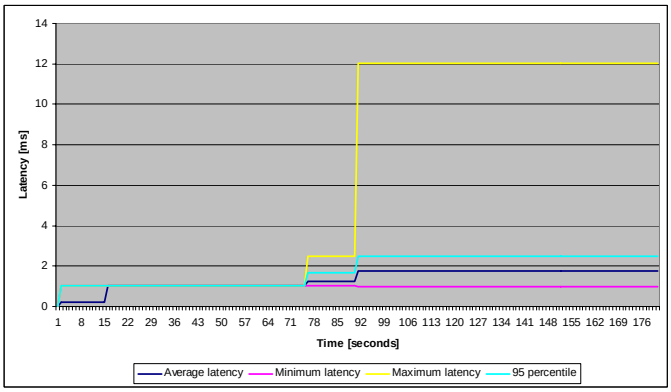

Figure 10. Delay results for voice when $\alpha_{Phase\,1} = 0.5$

Using the same scenario, the α_{Phase1} and α_{Phase2} variables were increased for the voice class to 0.6, so another 300 kb/s flow would be admitted. Figure 11 shows the throughput results, and it is easy to see that the system has now started to enter saturation. The best effort flows start suffering immediately, and minor disturbance in the voice throughput is shown in Figure 11.

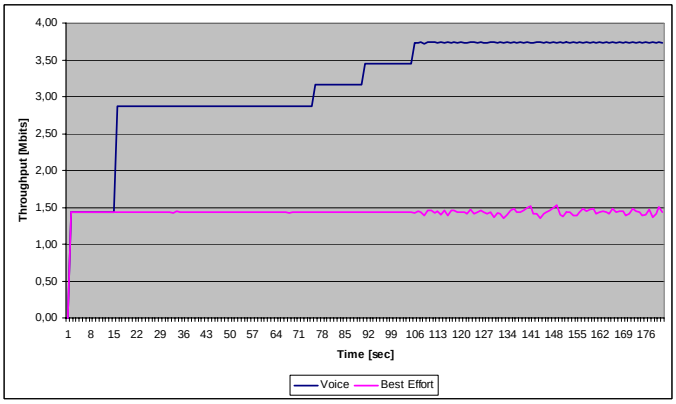

Figure 11. Throughput results when $\alpha_{Phase2} = 0.6$ for voice

The delay results when $\alpha_{Phase2} = 0.6$ for the voice class is shown in Figure 12. A rapid increase in the delay has clearly occurred, and the maximum delay has increased from 13 ms to 34 ms. The system is not in saturation yet, but it is in the middle stage between saturation and non-saturation.

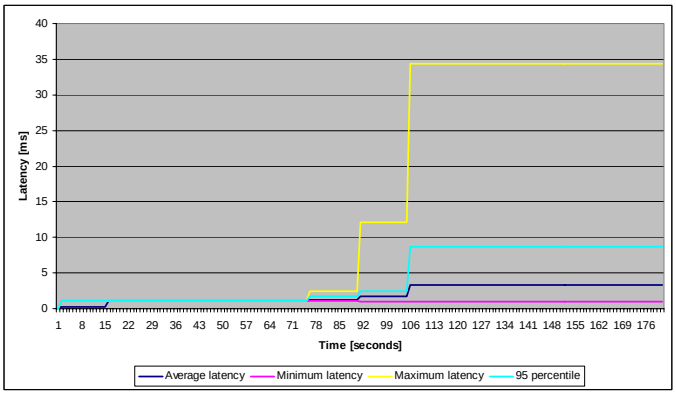

Figure 12. Delay results when $\alpha_{Phase2} = 0.6$ for voice

Alternatively, if the $\alpha_{Phase\ 1}$ for the best effort class was adjusted from 0.4 to 0.3, a different result is achieved as shown in Figure 13. The first five 1.5 Mb/s flows were all rejected, but four of the five 300 kb/s flows were accepted. The delay results for the best effort class stayed approximately the same as when $\alpha_{Phase1} = 0.4$.

Figure 13 also shows an interesting feature of Phase 1. In the first seconds of each new 1.5 Mb/s flow, the best effort class has around 150 kb/s in throughput. This is caused by the monitoring phase, and means that the flows passed Phase 1, but was quickly rejected in Phase 2. However, the quality of the already admitted flows was not aggravated as the figure shows. The delay results were not aggravated either. This is because Phase 1 takes this into account when accepting the flow, and therefore the flow only introduced a slightly overhead in Phase 2. This effect was explained in Section IV.

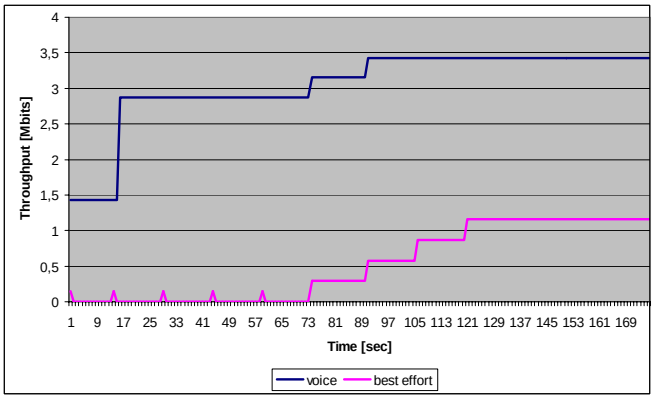

Figure 13. Delay results when $\alpha_{phase1} = 0.3$ for best effort

VI. Conclusion

This paper presents a model-based admission control which uses delay predictions and the utilization factor ρ_i as a threshold for transmissions. By using the medium access delay as a performance measure for admission control the queue lengths at each node are taken into account, and this is important for a satisfactory performance of the upper layer protocols. (Section III demonstrates the advantages of this approach over using throughput as the performance measure).

Our admission control algorithm is made up of two phases, where the first phase tries to predict the future. It is very difficult to achieve this with a good probability, and we have made several assumptions in our implementation of Phase 1. An example of a Phase 1 algorithm was presented.

Several simulation results were presented, with good throughput and delay results. By adjusting either α_{Phase1} or $\alpha_{Phase\ 2}$ or both, we were able to control how much traffic that was admitted. It is important that these variables are set according to the desired throughput level and delay level. This is especially true when the system is in the critical phase between non-saturation and saturation. A small increase of throughput could mean a high increase in delay.

Simulations showed that it was possible to omit Phase 1, but on behalf of the throughput, since $\alpha_{Phase\ 2}$ must be set extremely low. If $\alpha_{Phase\ 2}$ is set too high when disabling Phase 1, we saw that the throughput and delay would be severely aggravated, especially for the best effort class.

The paper also presents results of more general applicability. First, it demonstrates that it is possible to make measurements on the channel in such a way that other system parameters can be computed from an analytic model. Moreover, the monitoring in Phase 2 that might result in the rejection of flows can easily be applied as a general background process that works as a compliment to other admission control schemes.

VII. Future Work

A more complex Phase 1 algorithm should be devised, in order to justify more realistic scenarios. We have outlined several possible algorithms. A combination of several algorithms is the most promising approach.

It would also be of interest to evaluate what happens when Phase 1 is omitted. Moreover, assess how well Phase 2 performs as a standalone algorithm, by simply adjusting the different parameters.

VIII. References

1. IEEE, *Std 802.11*. 1999.
2. IEEE, *P802.11e/D13.0*. January 2005.
3. Yang Xiao, H.L., *Voice and Video Transmissions with Global Data Parameter Control for the IEEE 802.11e Enhanced Distributed Channel Access.* IEEE Trans. on parallel and distributed systems vol. 15, Nov. 2004.
4. Yang Xiao, H.L., Sunghyun Choi,, *Protection and Guarantee for Voice and Video Traffic in IEEE 802.11e Wireless LANs.* IEEE, 2004.
5. Yang Xiao, H.L., *Evaluation of Distributed Admission Control for the IEEE 802.11e EDCA.* IEEE Radio Communications, September 2004.
6. Daqing Gu, J.Z., *A New Measurement-Based Admisson Control Method for IEEE802.11 Wireless Local Area Networks.* IEEE, 2003.
7. Bai, A., et al., *A Class Based Dynamic Admitted Time Limit Admission Control Algorithm for 802.11e EDCA.* ASWN 2006, Berlin, 2006.
8. Liqiang Zhang, S.Z., *HARMONICA: Enhanced QoS Support with Admissin Control for IEEE 802.11 Contention-based Access.* IEEE RTAS, 2004.
9. Dennis Pong, T.M., *Call Admisson Control for IEEE 802.11 Contention Access Mechanism.* Globecom, 2003.
10. Yu-Liang Kuo, C.-H.L., Eric Hsaio-Kung Wu, Gen-Huey Chen, *An Admission Control Strategy for Differentiated Services in IEEE 802.11*. Globecom, 2003.
11. Zhen-ning Kong, D.H.K.T., Brahim Bensaou, *Measurement-assisted Model-based Call admission Control for 802.11e WLAN Contention-based Channel Access.* IEEE LANMAN, 2004.
12. Chun-Ting Chou, S.S., Kang G. Shin, *Achieving Per-Stream QoS with Distributed Airtime Allocation and Admission Control in IEEE 802.11e Wireless LANs.* IEEE, 2005.
13. Deyun Gao, J.C., King Ngi Ngan, *Admission Control in IEEE 802.11e Wireless LANs.* IEEE Network, August 2005.
14. Ergen, M.a.V., P., *Throughput analysis and admission control in IEEE 802.11a.* ACM-Kluwer MONET Special Issue on WLAN Optimization at the MAC and Network Levels, 2004.

15. Alizadeh-Shabdiz, F., Subramaniam, S., *Finite Load Analytical Model for the IEEE 802.11 Distributed Coordination Function MAC.* INRIA Sophia Antipolis, March 2003.
16. Cantieni, G., Ni, Q., Barakat, C. and Turletti, T., *Performance analysis under finite load and improvements for multirate 802.11*. Elsevier Comp. Comm. Journal, Vol. 28, Issue 10, pp. 1095-1109, June 2005.
17. Foh, C.H.a.Z., M., *Performance Analysis of the IEEE 802.11 MAC Protocol.* Proceedings of EW 2002, Florence, Italy pp. 184-190, Feb. 2002.
18. Malone, D.W., Duffy, K. and Leith, D.J., *Modelling the 802.11 Distributed Coordination Function with Heterogeneous Load.* Proceedings of Rawnet 2005, Riva Del Garda, Italy, April 2005.
19. Zaki, A.a.E.-H., M., *Throughput analysis of IEEE 802.11 DCF under finite load traffic*. First International Symposium on Control, Communications and Signal Processing, pp. 535-538, 2004.
20. Paal E. Engelstad, O.N.Ø., *Differentiation of Downlink 802.11e EDCA Traffic in the Virtual Collision Handler*. Proceedings of the 30th Annual IEEE Conf. on Local Computer Networks (LNC '05), WLN, Sydney, Australia, November 2005 (See also http://folk.uio.no/paalee/).
21. Paal E. Engelstad, O.N.Ø., *Analysis of the Total Delay of IEEE 802.11e EDCA and 802.11 DCF*. IEEE International Conference on Communication (ICC'2006), 2006 (See also http://www.unik.no/personer/paalee/).
22. Paal E. Engelstad, O.N.Ø., *The Delay Distribution of IEEE 802.11e EDCA and 802.11 DCF*. 25th IEEE International Performance Computing and Communications Conference (IPCCC'06), April 2006.
23. Kevin Fall, K.V., *The NS manual.* http://www.isi.edu/nsnam/ns/doc/index.html.
24. Sven Wietholter, C.H., *Design and Verification of an IEEE 802.11e EDCF Simulation Model in ns-2.26.* Technical University Berlin, Telecommunications Networks Group, November 2003.
25. Office, M., *Trendline in Excel.* http://office.microsoft.com/en-us/assistance/HP051984621033.aspx.